



This is Unit #4 of the BPEL Fundamentals course. In past Units we've looked at ActiveVOS Designer, Workspaces and Projects, so now let's take a look at the actual BPEL Processes.

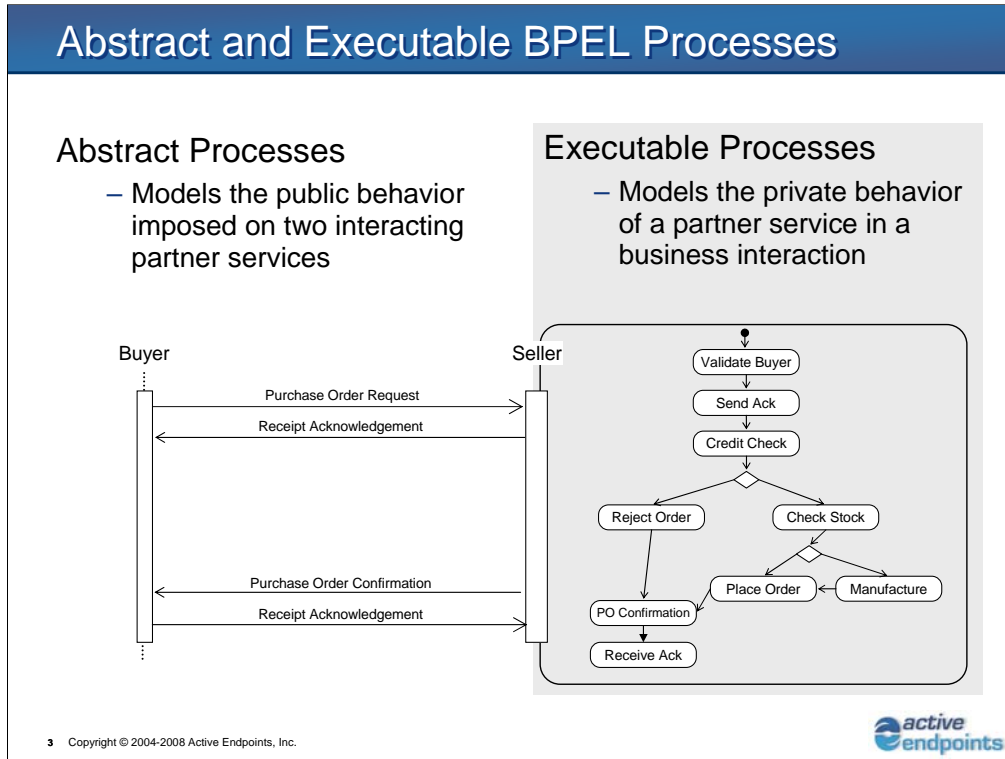
### Unit Objectives

- At the conclusion of this unit, you will be familiar with:
  - BPEL process definitions
  - Standard BPEL Faults
  - Working with BPEL processes
  - Considerations before getting started

2 Copyright © 2004-2008 Active Endpoints, Inc.



In this Unit we'll look at how BPEL Processes are defined and the standard faults supported by the system. Then we'll examine BPEL Processes and go over some things that should be considered before we get started.



On the left we see two partners, each with a role as a buyer or seller, and the traffic pattern of their interaction. This is an *Abstract Process* and it defines the process' public-facing behavior and is the conceptual definition of what will end up as the completed process. Think of it as the process from the view of a third party. This is not used much, mostly in the preliminary review of a design. The *Executable Process* on the right shows the private definition, and the actual interactions of the completed process. Think of it as the process when viewed from behind the developer's curtain. So, on the right we see the actual implementation of the process that will be deployed. From now on, we are only concerned with executable processes.

## BPEL process Overview and Syntax

- Represents the definition of a business process
  - Is defined in an XML format that conforms to the WS-BPEL 2.0 Executable XML Schema
  - Defines the variables, handlers, partner links, and a single primary activity

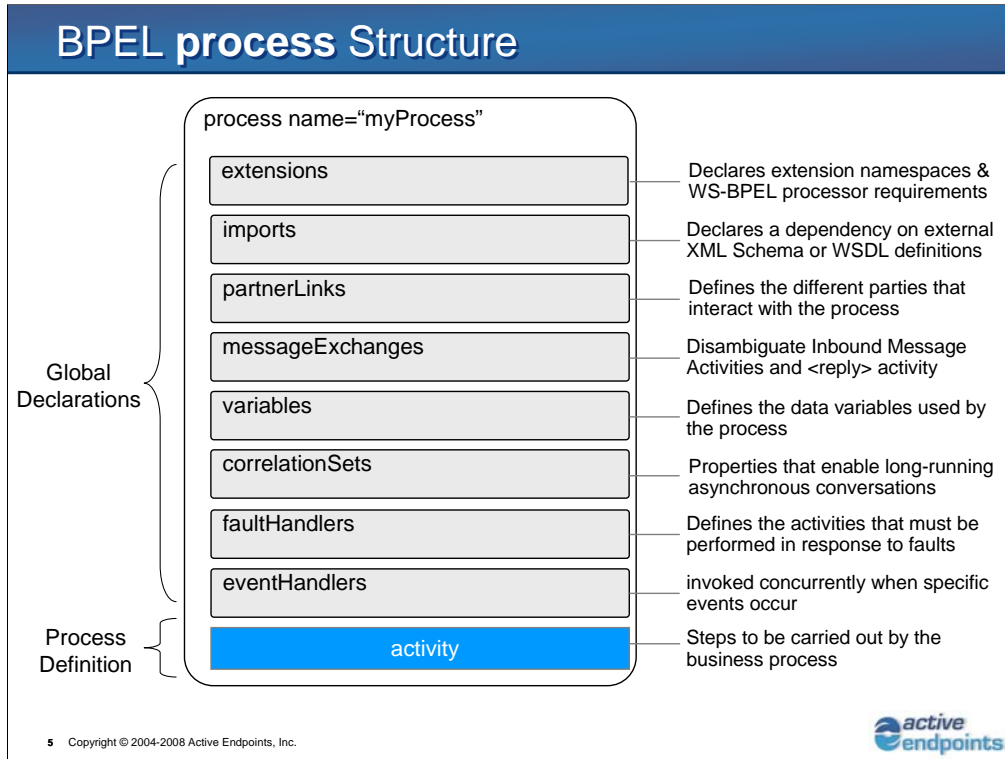
```
<process name="NCName" targetNamespace="anyURI"
  queryLanguage="anyURI"?
  expressionLanguage="anyURI"?
  suppressJoinFailure="yes|no"?
  exitOnStandardFault="yes|no"?
  xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable">
  ...
</process>
```

4 Copyright © 2004-2008 Active Endpoints, Inc.



A BPEL Process is one that defines a business process, is displayed in XML format and conforms to the BPEL 2.0 schema. The Process file contains the definitions listed on the slide, including:

- Process Variables
- Handlers (Event, Fault, Termination, Compensation)
- Partner Links (i.e., instances of the WSDL-defined PartnerLinkTypes)
- A single primary activity (Start Activity) that will instantiate the process.



Here we see the structure of a process. At the top we name the process, then add the process' global declarations and follow up with the actual process definition. The global declarations section includes the extensions, imports, partnerLinks, etc. that are used in the process. The process definition has the actual activities that make up the process' functionality.

## BPEL process Example


```

<process name="ExampleProcess" targetNamespace="urn:Example"
  xmlns="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
  xmlns:c="urn:Calc">
  <partnerLinks>
    <partnerLink myRole="calculator" name="calcServicePL"
      partnerLinkType="c:calcServicePLT" />
  </partnerLinks>
  <variables>
    <variable messageType="c:CalcInput" name="CalcInput" />
    <variable messageType="c:CalcOutput" name="CalcOutput" />
  </variables>
  <sequence>
    <receive createInstance="yes" operation="CalcOperation"
      partnerLink="calcServicePL" portType="c:CalcPT"
      variable="CalcInput" />
    <assign>
      <copy>
        <from part="Input" query="xpath" variable="CalcInput" />
        <to part="Output" query="xpath" variable="CalcOutput" />
      </copy>
    </assign>
    <reply operation="CalcOperation" partnerLink="calcServicePL"
      portType="c:CalcPT" variable="CalcOutput" />
  </sequence>
</process>

```

Global  
Declarations

Process  
Definition



6 Copyright © 2004-2008 Active Endpoints, Inc.

We assemble the process graphically in ActiveVOS Designer, but here is the actual BPEL code that's created in the background. It follows the file structure we saw on the previous slide:


- It names the process and designates the target namespace
- It lists the global declarations, which define items such as PartnerLinks and process variables
- It ends with the process definition, which is the process' individual activities and containers that make up the process' actions.

In this example the actual process boils down to: An initial Receive, an Assign with a single Copy operation, and a Reply.

## BPEL Activities

- Activities represent the actions to be carried out within a business process
- BPEL defines two types of activities
  - Basic Activities
    - invoke
    - receive
    - reply
    - assign
    - compensate
    - compensateScope
    - empty
    - exit
    - throw
    - rethrow
    - validate
    - wait
  - Structured Activities
    - flow
    - forEach
    - if
    - pick
    - repeatUntil
    - scope
    - sequence
    - while

7 Copyright © 2004-2008 Active Endpoints, Inc.



BPEL *activities* are the actions the process actually executes. Basic activities are those that are considered to be atomic, meaning that they cannot be subdivided. Structured activities are those activities which can contain other activities.

## Basic and Structured Activities

- Basic activities are combined using structured activities

**Program-like structure**  
Control flow defined by sequences, loops and branches

sequence

basic activity

loop (while condition is true)

basic activity

basic activity

if

condition 1	condition 2	else
basic activity	basic activity	basic activity

**Graph-like structure**  
Control flow defined by a network of links

flow

basic activity

structured activity

basic activity

basic activity

basic activity

8 Copyright © 2004-2008 Active Endpoints, Inc.

There are two major ways to define a business process – using a program structure and using a graph structure. A program structure is defined by its content, is controlled by loops, logic, branches, sequences and executes in a linear fashion. A graph structure is in a format defined by a network of links and it can execute its activities in parallel. These two approaches can be mixed and matched as needed. Why have both? Recall that BPEL originated as the merging of two different specifications - the program structure from Microsoft's XLang and the graph structure that is from IBM's WSFL.

Note that BPEL's Flow activity allows for the parallel processing of process activities (the only such activity, other than the ForEach loop.)

## Primary Activity

- A BPEL process definition contains a single primary activity
  - The main entry point for the process definition
- Primary activity can be
  - Simple basic activity
  - Complex structured activity
    - With many nested activities contained to arbitrary depth
- Meaningful BPEL processes use either the **sequence** or **flow** activity as the primary activity

© Copyright © 2004-2008 Active Endpoints, Inc.



Recall that a BPEL process *must* receive a message in order to instantiate the process. All BPEL processes *must* have a single primary activity, which receives the message and is the main entry point of the process. This primary activity can be a simple activity or a structured activity. If it is a structured activity it can be nested to any depth. Most BPEL processes will have a Sequence or a Flow as their primary activity.

## Common Activity Syntax

### ■ Common Attributes (Optional)

```
name = "ncname" ?
suppressJoinFailure = "yes | no" ?
```

### ■ Common Elements (Optional)

```
<targets>?
  <joinCondition expressionLanguage="anyURI"??>
    bool-expr
  </joinCondition>
  <target linkName="NCName" />+
</targets>
<sources>?
  <source linkName="NCName">+
    <transitionCondition expressionLanguage="anyURI"??>
      bool-expr
    </transitionCondition>
  </source>
</sources>
```

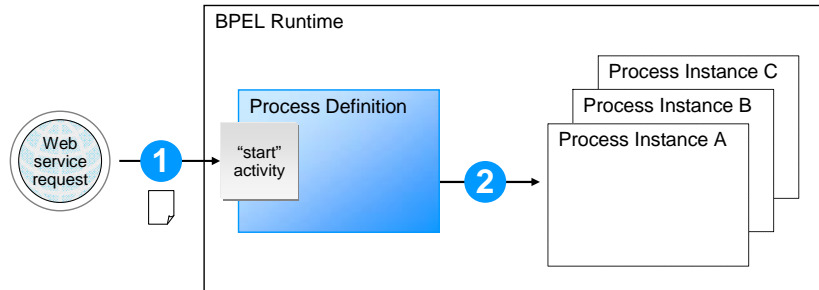
10 Copyright © 2004-2008 Active Endpoints, Inc.



Every activity also can have a set of common attributes and elements. Many of these elements will only apply to certain activities. For example, Flow activities will have links, joinConditions and source/target designations.

## BPEL Process Life Cycle - Start

- Creation of a process instance is always implicit
  - Instances are created upon receiving a message targeted for a “start” activity
  - Executable processes must have at least one “start” activity



11 Copyright © 2004-2008 Active Endpoints, Inc.



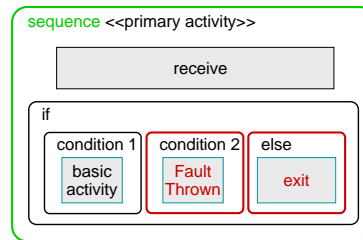
Now let's take a look at the life cycle of a BPEL process.

1. ) When an incoming message is received, the server will try and match it (i.e., correlate it) to a currently running instance.
2. ) If it cannot do so, it will create a new process instance, and the message will be received by this new instance.

Remember that every executable process **MUST** have at least one “start” activity.

## BPEL Process Life Cycle - Terminate

- Termination of a process instance occurs
  - **Normally**, when primary activity of process completes
  - **Abnormally**, when
    - Unhandled fault reaches the primary activity
    - Explicitly terminated using an **exit** activity



12 Copyright © 2004-2008 Active Endpoints, Inc.



Now that we know how a process is instantiated, let's look at how a process terminates. There are two ways a running process can terminate – normally or abnormally.

*Normal termination* is when the process activities are completed successfully.

*Abnormal termination* is when the process activities are not completed successfully, usually because of a fault being thrown that reaches the process level without being handled, or through the use of an “Exit”, which stops the running process instantly.

Note that the Exit activity is new to the BPEL 2.0 specification and was previously known as the Termination activity in BPEL v1.1

### Unit Objectives

- At the conclusion of this unit, you will be familiar with:
  - ✓ BPEL process definitions
  - Standard BPEL Faults
  - Working with BPEL processes
  - Considerations before getting started

13 Copyright © 2004-2008 Active Endpoints, Inc.



Now that we know how the process works, let's take a look at some standard BPEL faults.

### Standard BPEL Faults


- BPEL defines a set of built-in system-centric faults
  - BPEL runtime is responsible for signaling these faults
  - Defined in the BPEL namespace (bpel:)
    - <http://docs.oasis-open.org/wsbpel/2.0/process/executable>

14 Copyright © 2004-2008 Active Endpoints, Inc.




The ActiveVOS Engine generates and manages all system faults, and it signals the process once a fault has been encountered. These are the faults that are defined in the BPEL executable namespace shown here.

Standard BPEL Faults	
Fault Name	Fault is thrown when:
ambiguousReceive	<b>more than one IMA for the same partnerLink, portType, operation but different correlationSets, matches an incoming request message</b>
completionConditionFailure	<b>if it is determined that the completion condition of a forEach activity can never be true.</b>
conflictingReceive	<b>more than one receive activity or equivalent are enabled simultaneously for the same partner link, port type, operation and correlation set(s)</b>
conflictingRequest	<b>more than one synchronous inbound request on the same partner link for a particular port type, operation and correlation set(s) are active</b>

15 Copyright © 2004-2008 Active Endpoints, Inc. 

On the following four slides are the Standard BPEL faults. These are faults that are generated by the system.

Standard BPEL Faults (cont.)	
Fault Name	Fault is thrown when:
correlationViolation	the contents of the messages that are processed in an invoke, receive, or reply activity do not match specified correlation information
invalidBranchCondition	the integer value used in the <branches> completion condition of <forEach> is larger than the number of directly enclosed <scope> activities
invalidExpressionValue	an invalid value is returned from the execution of an expression
invalidVariables	an XML Schema validation (implicit or explicit) of a variable value fails
joinFailure	join condition of an activity evaluates to false
mismatchedAssignmentFailure	incompatible types are encountered in an assign activity

16 Copyright © 2004-2008 Active Endpoints, Inc. 

.. And some more faults.

## Standard BPEL Faults (cont.)

Fault Name	Fault is thrown when:
missingReply	<b>a scope (or process) completes without replying to a receive</b>
missingRequest	<b>a reply activity is executed with no open receive</b>
scopeInitializationFailure	<b>there is any problem creating any of the objects (e.g., variables or partneLinks) defined as part of scope initialization</b>
selectionFailure	<b>a selection operation performed either in a function or in an assignment, encounters an error</b>
subLanguageExecutionFault	<b>an unhandled exception is encountered during the execution of a expression or query</b>
uninitializedPartnerRole	<b>an invoke or assign activity references a partner link whose partnerRole endpoint reference is not initialized</b>

17 Copyright © 2004-2008 Active Endpoints, Inc.



... and some more!

## Standard BPEL Faults (cont.)

Fault Name	Fault is thrown when:
uninitializedVariable	<b>there is an attempt to access the value of an uninitialized part in a message variable</b>
unsupportedReference	<b>unable to interpret the combination of the reference-scheme attribute and the content element OR just the content element alone</b>
xsltInvalidSource	<b>if a invalid XML element is provided as the source document to the <code>bpel:doXsltTransform</code> function call</b>
xsltStylesheetNotFound	<b>when the named style sheet in a <code>bpel:doXsltTransform</code> function call was not found</b>

18 Copyright © 2004-2008 Active Endpoints, Inc.



... and just a few more. That's all of the Standard BPEL Faults, but of course, there are also the faults that are defined by the process, too. There are an *unlimited* number of these User-defined faults.

### Unit Objectives

- At the conclusion of this unit, you will be familiar with:
  - ✓ BPEL process definitions
  - ✓ Standard BPEL Faults
  - Working with BPEL processes
  - Considerations before getting started

19 Copyright © 2004-2008 Active Endpoints, Inc.




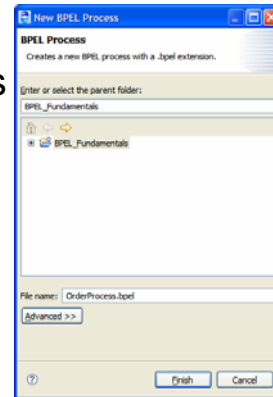
Now that we've looked at BPEL process definitions and the standard system faults, let's see how to work with BPEL processes.

## Create a BPEL Process

1. Select File>New>BPEL Process
  - New BPEL Process Wizard opens
2. Specify which parent folder to add the process to
3. Specify a name for the new process
4. Select Finish
  - Process editor opens

Alternatively either:

- Right click on a project folder, then New>BPEL Process
- Select New  toolbar dropdown, then BPEL Process

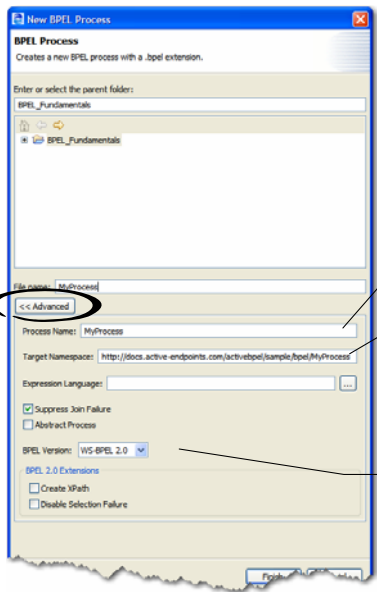


20 Copyright © 2004-2008 Active Endpoints, Inc.



In ActiveVOS Designer, we have already created a new workspace and a new project, so now we need to create the actual process. Go to File->New->BPEL Process and when the Wizard opens, specify what folder to store the new file in, what you want to name it, and then click finish.

### Create a BPEL Process – Advanced Options




The screenshot shows the 'New BPEL Process' dialog box. It has a title bar 'New BPEL Process' and a close button. The main area is titled 'BPEL Process' and contains the following fields and options:

- 'Enter or select the parent folder:' with a tree view showing 'BPEL\_Fundamentals'.
- 'Process Name:' with the value 'MyProcess'.
- 'Target Namespace:' with the value 'http://docs.active-endpoints.com/activevop/sample/Bpel/MyProcess'.
- 'Expression Language:' with a dropdown menu.
- Checkboxes for 'Suppress Join Failure' (checked), 'Abstract Process' (unchecked), 'Create XPath' (unchecked), and 'Disable Selection Failure' (unchecked).
- 'BPEL Version:' with a dropdown menu set to 'WS-BPEL 2.0'.
- 'BPEL 2.0 Extensions' section with the same checkboxes as above.

Three callout boxes point to specific elements:

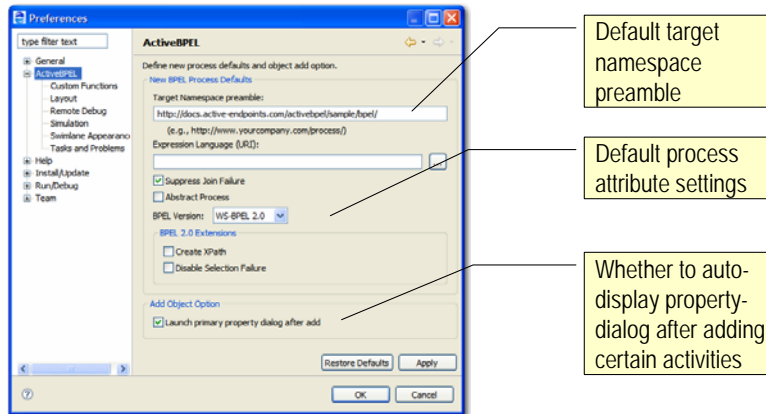
- The first callout points to the 'Process Name' field and contains the text: 'The same as the file name'.
- The second callout points to the 'Target Namespace' field and contains the text: 'Uses the Target namespace preamble workspace preference combined with the file name'.
- The third callout points to the 'BPEL 2.0 Extensions' section and contains the text: 'Uses the values defined in the workspace preferences'.

21 Copyright © 2004-2008 Active Endpoints, Inc. 

When you have the New BPEL Process dialog open, you can click on Advanced button to change the preamble for the target nameSpace. When we change the target namespace preamble the new text will be amended to the nameSpaces created by ActiveVOS Designer. If we do nothing, it will use what's in the preferences by default.

## ActiveVOS Designer Preferences

- Default preferences used for creating new processes
  - Can override preferences on a per process basis



22 Copyright © 2004-2008 Active Endpoints, Inc.



While we are here, we can change the system's default preferences, as needed. Note that these settings are at the Workspace level, but we can override these defaults on a per-process basis. The namespace preamble is pre-pended to the namespace for any processes that you create after changing the value.

## Importing an Existing BPEL Process

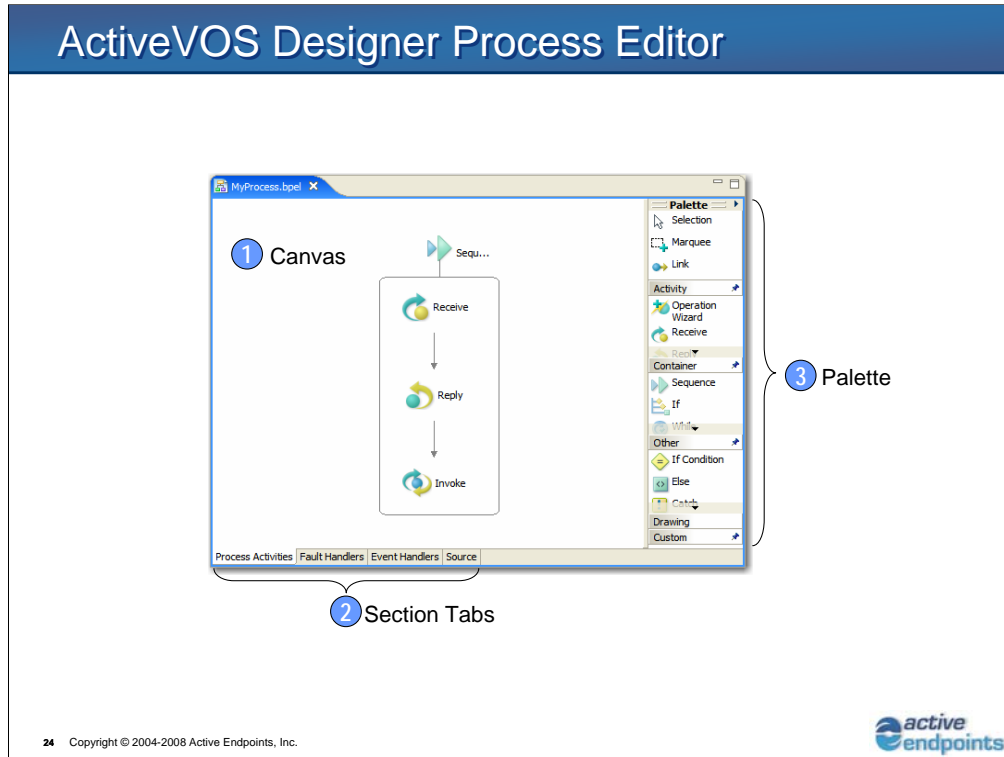
- You can import existing BPEL process into your project
  - e.g., you have created a BPEL process outside of ActiveVOS Designer
- Steps
  - In the Project Explorer, right-click the parent folder and then select **Import...**
  - Select **File System** and then select **Next**
  - Browse to the appropriate location on your file system and then select the resources to import

*Note: You must also add any additional WSDL and Schema files referenced in your BPEL process to the Project Explorer*

23 Copyright © 2004-2008 Active Endpoints, Inc.



You can import individual resource files or entire processes into your project. Open up the Project Explorer, use the Right Mouse menu on the parent folder, select “Import”, select File System, click Next and then browse to the location. Note that if you import a process you *must* also import the WSDL and Schema files referenced by that process.

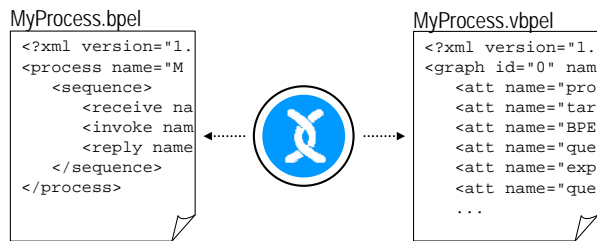


Here is the ActiveVOS Designer's Process Editor.

- 1.) The Process Editor's **Canvas** is the default Editor for the tool.
- 2.) By default, the Process Editor's canvas opens to the Process Activities tab, from among the **Section** tabs.
- 3.) The **Palette** has all of the BPEL activities, as outlined by the BPEL 2.0 specification, and its presentation can be customized.

## .vbpel File

- For every .bpel file there is a corresponding .vbpel file that stores the process layout
  - Any information that is not standard BPEL is placed in the .vbpel file
    - Activity placement, drawing elements (e.g., ovals, labels)
  - An XML file, based on the eXtensible Graph Mark-up and Modeling Language (XGMML) standard



25 Copyright © 2004-2008 Active Endpoints, Inc.



For every “.bpel” file there is a corresponding “.vbpel” file to accompany it. The “.bpel” file contains the process description in XML format and contains everything needed to deploy and execute the process. The “.vbpel” (pronounced VEE-BEEP-UHL) is an extra file that represents the visual layout of the process' diagram. This file is in XGMML format and contains everything needed to display the process in a graphical format. This file is not needed for deployment because a deployed process does not usually need to be displayed anywhere.

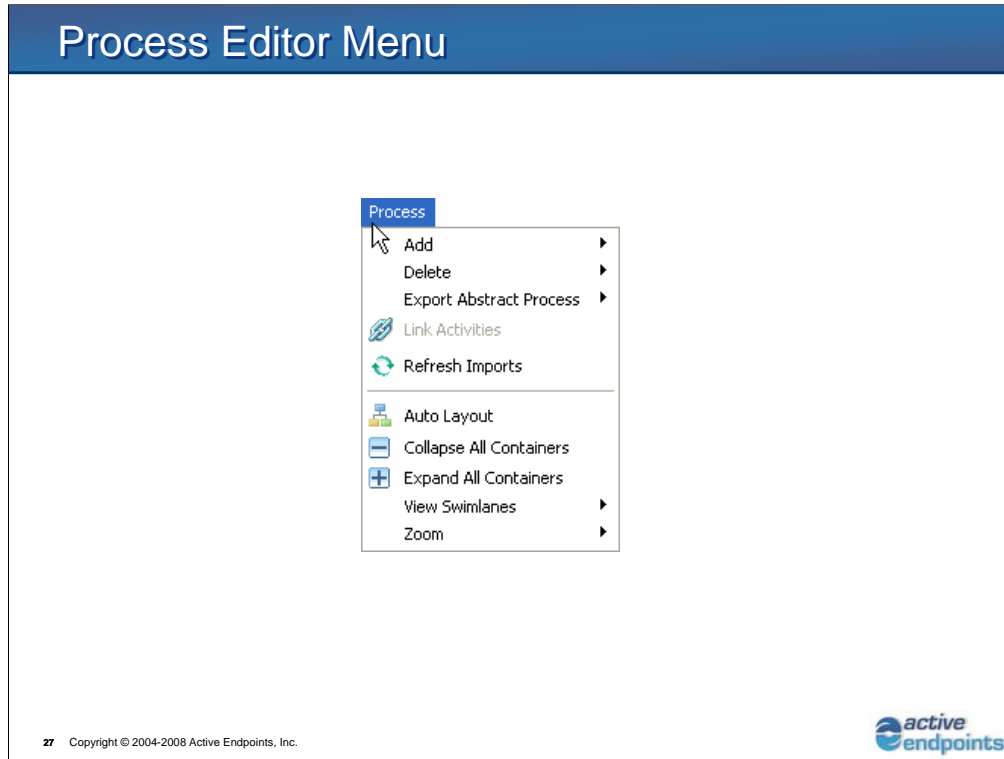
## Showing .vbpel Files in Project Explorer

- By default, .vbpel files are excluded from display in the Project Explorer
  - To display .vbpel files in the Project Explorer
    - select Filters... from the Project Explorer menu, then **uncheck** the \*.vbpel entry
- You may need to use .vbpel files for the one of the following reasons
  - When copying BPEL files
    - .vbpel files do not get copied
  - When using Replace with Local History
    - Need to separately replace the .vbpel file
  - When editing the BPEL file outside ActiveVOS Designer
    - .vbpel file will need to be re-generated

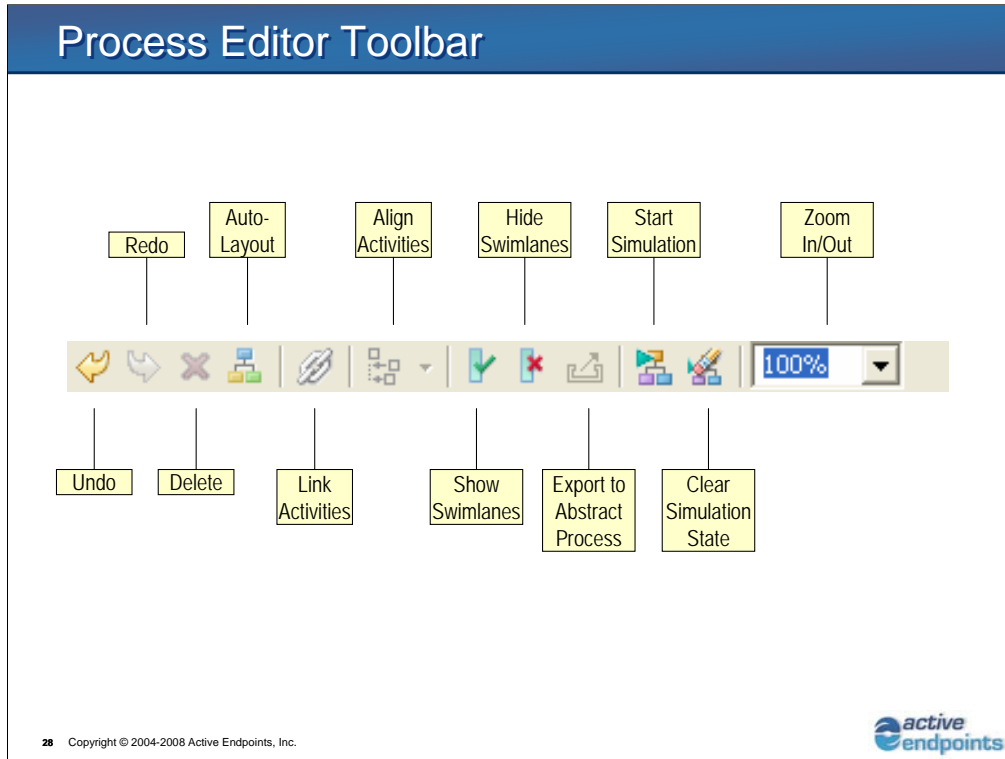
26 Copyright © 2004-2008 Active Endpoints, Inc.



When you copy the project files, the .vbpel is not copied automatically. You have to do that yourself. Even after you copy it, it is filtered out of the Project Explorer by default. Use "Project Explorer->Customize View" and then *uncheck* the box next to the "\*.vbpel" type to see them in the heirarchy.



Clicking on the process editor and opening the Right Mouse menu gives you this set of choices. It supports all of the common functionality of the menus and other Right Mouse actions. As well as display choices, such as Collapsing or Expanding containers and whether to show Swimlanes.



Toolbar options provide much of the same functionality...

## Working with Activities


- To add an activity to the process
  - First choose the appropriate activity from the palette and then click within the process editor canvas
    - Alternatively, drag and drop an activity from the palette to the desired location in the process editor canvas
- You can also
  - Copy and paste activities
  - Select multiple activities and update their properties as a group
  - Align multiple activities with one of the align options from the process toolbar

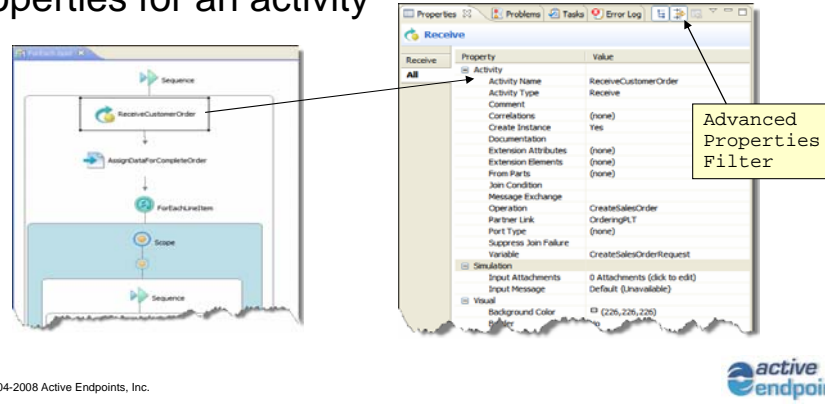
29 Copyright © 2004-2008 Active Endpoints, Inc.



To create a new activity, select the activity in the palette which “loads” the activity, and then click on the process where you’d like the activity to be inserted. Alternatively, you can use Drag & Drop to insert an activity from the palette onto the canvas. You can also use Copy and Paste with activities that are already in the diagram, and you can select multiple activities and then edit their properties as a group. Finally, you can Align activities in the graphic as needed, using “AutoLayout” or “Align Left,” for example.

## Activity Properties

- With the activity in focus on the process editor canvas, you can set property values in the Properties view 
- Select the Advanced button to display all the properties for an activity



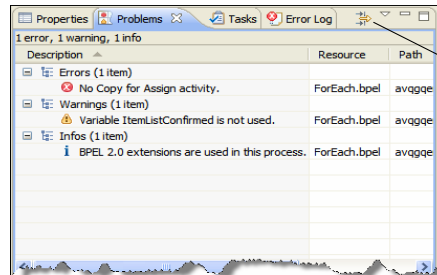
30 Copyright © 2004-2008 Active Endpoints, Inc.



Select an activity on the drawing canvas and then select the Properties View. Note that you may have to use the Advanced Properties button to see all of the activity's properties.

## Static Analysis

- When you save your process, static analysis is performed on the BPEL file
- Validates the BPEL definition, XPath syntax, and WSDL references
  - Adds items for any issues found to the Problems view



Can filter items to view only warnings and errors associated with a particular resource or group of resources

31 Copyright © 2004-2008 Active Endpoints, Inc.



Static Analysis is a syntax and requirements check, which verifies that the BPEL process is consonant with the definitions required by the BPEL 2.0 specification. This check is done whenever you save your files. The results show up on the problems tab and you can filter for which elements you want reports on, and for errors/warnings/information, as needed. For more details on Static Analysis you can go to Appendix B of the BPEL specification, which covers all 95 checkpoints for a BPEL process.

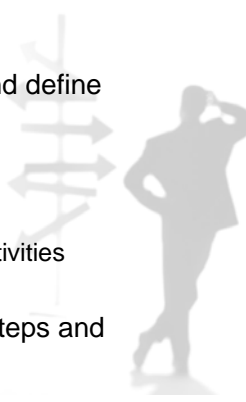
## Unit Objectives

- At the conclusion of this unit, you will be familiar with:
  - ✓ BPEL process definitions
  - ✓ Standard BPEL Faults
  - ✓ Working with BPEL processes
  - Considerations before getting started


Now, let's take a look at some things to consider before we start working.

## Process Design Approaches

- **Top-Down**
  - Sketch out the high-level intent of the business process
    - Use BPEL constructs to model generic process steps and business rules
  - Derive implementation requirements
    - Define message definitions and operations
  - Bind BPEL constructs to implementation details and define information flow
- **Bottom-Up**
  - Start with existing implementation details
    - Valid service interface definitions for process and activities
    - Minimally have message definitions
  - Use BPEL constructs to model concrete process steps and business rules and define information flow



33 Copyright © 2004-2008 Active Endpoints, Inc.



There are two different approaches to designing a BPEL process - the top-down approach and the bottom-up approach.

If using the top down approach:

- sketch out your high level intent with BPEL constructs
- derive your implementation requirements – operations and messages
- fill in the details of the information flow
- build your process implementation constructs

If using the bottom up approach:

- create your activities and process definition
- Add message definitions
- tie them together to create your information flows and process steps based on your business rules

## Process Design Planning

### ■ Identify

- Business process to automate
  - e.g., Purchase Orders, Claims, Service Provisioning
- Discrete business functions that are described as Web services
  - e.g., Inventory, credit rating, shipping and logistics
- Information flow between the process steps
  - e.g., Receive PO request as input to process, pass the list of order items to the check inventory levels, and so on
- Business rules required to orchestrate the process
  - e.g., After checking inventory levels, depending on the result, either place the order or send an out-of-stock notification

34 Copyright © 2004-2008 Active Endpoints, Inc.



Before you begin creating your process, though, you need to step back and plan the process design.

- 1.) What is the Process and what is its purpose? What does it actually do?
- 2.) What Business Functions (i.e., Web Services) will the process will use? Do they all exist now or do they have to be created?
- 3.) What is the Information flow (i.e., the variables used) both inside and between the process steps? //this can also involve *mapping* data between partner invocations.
- 4.) What are the Business rules (logic/flow/execution routes) that must be applied to the process? How do they affect the way you will orchestrate the process?

## Unit Summary

- Now you are familiar with:
  - BPEL process definitions
  - Standard BPEL Faults
  - Working with BPEL processes
  - Considerations before getting started